1. In a voice-extensible-markup-language-enabled voice-application deployment architecture, an application logic for determining which portions of a voice application for deployment should be cached at an application-receiving end system or systems, comprising:

a processor for processing the voice application according to sequential dialog files of the application;

a static content optimizer connected to the processor for identifying files containing static content; and

a dynamic content optimizer connected to the processor for identifying files containing dynamic content;

characterized in that the optimizers determine which files should be cached at which end-system facilities, tag the files accordingly, and prepare those files for distribution to selected end-system cache facilities for local retrieval during consumer interaction with the deployed application.

2. The application logic of claim 1 wherein the static and dynamic optimizers are software routines.

3. The application logic of claim 1 wherein the static and dynamic optimizers are firmware components embedded into the processor.

4. The application logic of claim 1 wherein the processor is a dialog runtime processor dedicated to processing subsequent dialogs of a voice application.

5. The application logic of claim 1 wherein the deployment architecture includes an application server and a voice portal.

6. The application logic of claim 1 wherein the dynamic optimizer identifies dynamic content according to a determination of non-recurring menu dialog and non-recurring result dialog fetched as a result of consumer interaction with the voice application.

7. The application logic of claim 1 wherein the cache facility at the end system is a telephony server cache.

8. The application logic of claim 1 wherein the cache facility at the end system is a Web controller cache.

9. The application logic of claim 1 wherein the file tagging is accomplished using HTTP 1.1 resource tagging.

10. The application logic of claim 1 wherein dynamic tagging by the dynamic optimizer uses results from statistical analysis to determine which files to tag for distribution to an end-system cache.

11. The application logic of claim 1 wherein dynamic optimization continues after application deployment, the continued dynamic tagging relying on changing statistical probability results.

12. A system for creating and distributing interactive voice applications to end users comprising:

a voice application server;

a voice application;

a voice portal; and

a network for delivery;

characterized in that the voice application determines which dialog files of a finished voice application will be cached locally at the voice portal for subsequent local retrieval during end-user interaction with the application.

13. The system of claim 12 wherein the voice application has a static and dynamic optimizer connected to a dialog runtime processor, the optimizers cooperating locally to tag and prepare cacheable content of the voice application for caching and subsequent retrieval from the voice portal.

14. The system of claim 12 wherein the network for delivery is a telephony network.

15. The system of claim 12 wherein the network for delivery is a data network.

16. The system of claim 12 wherein the delivery network is a combination of a data network and a telephony network the application delivered through a network bridge.

17. The system of claim 13 wherein the static and dynamic optimizers are firmware components embedded into the processor.

18. The system of claim 13 wherein the dialog runtime processor is dedicated to processing subsequent dialogs of a voice application.

19. The system of claim 13 wherein the dynamic optimizer identifies dynamic content according to a determination of non-recurring menu dialog

and non-recurring result dialog fetched as a result of consumer interaction with the voice application.

20. The system of claim 12 wherein the voice portal includes a telephony server and cache.

21. The system of claim 20 further including a Web controller and cache.

22. The system of claim 13 wherein the static and dynamic optimizers tag files determined to be cacheable according to HTTP 1.1 regimen.

23. The system of claim 22 wherein dynamic tagging by the dynamic optimizer uses statistical analysis to determine which files to tag for distribution to an end-system cache.

24. The system of claim 13 wherein dynamic optimization continues after application deployment, the continued dynamic tagging relying on changing statistical probability results.

25. A method for identifying specific dialog files of a voice application for local file caching at targeted end systems, the application pending deployment from a voice application server and deploying the selected files to the targeted cache systems for local retrieval during voice application interaction comprising steps of:

(a) running the voice application at the voice application server;

(b) identifying static dialogs of the application and tagging them appropriately;

(c) identifying dynamic dialogs of the application and tagging them appropriately;

(d) deploying the static and dynamic dialog files identified and tagged to selected target cache systems; and

(e) retrieving, at the end systems, the tagged files from local cache to play in real time and in proper order with the deployed voice application.

26. The method of claim 25 wherein in step (a) the application is run on a runtime processor connected to a rules engine.

27. The method of claim 25 wherein in step (b) the static dialogs are identified and tagged by a static optimizer routine connected to the processor.

28. The method of claim 25 wherein in step (c) the dynamic dialogs are identified and tagged by a dynamic optimizer routine connected to the processor.

29. The method of claim 25 wherein in steps (b) and (c) tagging is accomplished using HTTP 1.1 regimen.

30. The method of claim 25 wherein in step (d) the selected files are deployed ahead of the voice application, the deployed application, when deployed, missing the selected files.

31. The method of claim 25 wherein in step (d) the selected files are deployed with the voice application and saved to the local cache systems at a first interaction with the deployed application.

32. The method of claim 25 wherein in step (c) dynamic dialogs include dynamic menus and dynamic data results fetched as a result of menu interaction.